4 Quantifying Networks

Networks: An Introduction,
M. E. J. Newman, Oxford University Press (2010), Chapter 7 (Also see 11).

Complex networks: Structure and dynamics,
Boccaletti *et al.*, Physics Reports 424 (2006), Section 7.

4.1 Basic measures

4.1.1 Node Degree

The node degree is also sometimes called the *degree centrality*, as it gives an idea of the relative importance (or *centrality*) of a node compared the others in the network. The degree of node i is given by:

$$k_i = \sum_{j=1}^{N} A_{ij}. \tag{1}$$

For directed networks this expression give the *in-degree*: $k_i^{\text{in}} = \sum_{j=1}^{N} A_{ij}$. For the *out degree* we sum over columns instead of rows: $k_j^{\text{out}} = \sum_{i=1}^{N} A_{ij}$, which is the same as transposing the adjacency matrix: $k_i^{\text{out}} = \sum_{j=1}^{N} A_{ij}^T$

4.1.2 Edge Density

The edge density is the ratio of the actual number of edges to the total possible number $\frac{1}{2}N(N-1)$. If m is the number of edges then each edge has $2m$ ends, which is also the sum of all node degrees:

$$\begin{aligned} 2m &= \sum_{i=1}^{N} k_i && (2)\\ \text{or} & & & \\ m &= \frac{1}{2}\sum_{ij} A_{ij} \quad \left(m = \sum_{ij} A_{ij} \text{for directed} \right). && (3) \end{aligned}$$

The mean degree is:

$$\bar{k} = \frac{1}{N}\sum_{i=1}^{N} k_i = \frac{2m}{N}. \tag{4}$$

Therefore the network edge density is:

$$\rho = \frac{2m}{N(N-1)} = \frac{\bar{k}}{N-1}. \tag{5}$$

We've also already seen a number of measure such as the number of length r paths between nodes i and j: $n_r = [A^r]_{ij}$, loops of length r: $L_r = [A^r]_{ii} = \text{Tr}A^r$ and the transitivity $c = \dfrac{\text{Tr}A^3}{\sum\limits_{i \neq j} A_{ij}^2}$.

4.2 Centrality Measures

Centrality is a measure of the importance of a node or an edge relative to others in the network.

4.2.1 Betweenness Centrality

This measures the relative number of geodesic paths passing through a node:

$$x_i = \frac{1}{N^2} \sum_{st} \frac{n^i_{st}}{g_{st}}, \tag{6}$$

where n^i_{st} is number of geodesic shortest paths between s and t passing through node i and g_{st} is total number of geodesic paths between s and t. This can also be applied to edges instead of nodes, giving the *"edge betweenness centrality"*.

4.2.2 Eigenvalue Centrality

If we consider a node to be important not only due to the number of its links (its degree) but also the importance of its network neighbours then we can define another measure. This can be calculated iteratively from an initial guess at the importance (either the degree or just all $x_i = 1$ will do. At each step we recalculate the centrality measure:

$$x'_i = \sum_j A_{ij} x_j, \tag{7}$$

or in matrix form:

$$\mathbf{x}' = A\mathbf{x}. \tag{8}$$

This is repeated until it converges suitably on some value:

$$\mathbf{x(t)} = A^t \mathbf{x(0)}. \tag{9}$$

This is recognised as the *power method* for calculating largest (most positive) eigenvalue of a matrix and its corresponding eigenvector. Indeed the above centrality problem can also be stated that we wish to find some values of x_i such that each node has a centrality that is proportional to the sum of its neighbours' values:

$$x_i = \lambda^{-1} \sum_j A_{ij} x_j, \tag{10}$$

where λ^{-1} is a constant of proportionality. In matrix form this is the eigenvalue equation:

$$A\mathbf{x} = \lambda\mathbf{x}. \tag{11}$$

The *Perron-Frobenius theorem* tells us that (9) converges to the eigenvector $\mathbf{x}$ with all positive values corresponding to the largest eigenvector λ_N, which is positive and real. Hence the vector $\mathbf{x}$ gives us the required centrality measure.

For directed networks (e.g. web pages and links) we choose the right eigenvectors so that we count incoming links:

$$x_i = \lambda_R^{-1} \sum_j A_{ij} x_j, \quad A\mathbf{x} = \lambda_R \mathbf{x} \tag{12}$$

4.2.3 Modified eigenvalue centrality measures

The above method can have problems in cases where nodes which are very highly cited by low-value nodes no not obtain enough centrality weight in the network, and there are also problems with acyclic graphs (e.g. citation networks) [See Newman]. This can be rectified in various ways.

***Katz centrality*:** We can add a small constant centrality to each node to prevent the above problem:

$$x_i = \alpha \sum_j A_{ij} x_j + \beta, \quad \mathbf{x} = \alpha A \mathbf{x} + \beta \mathbf{1}, \tag{13}$$

where α, β are positive constants. setting $\beta = 1$ this rearranges to:

$$\mathbf{x} = (I - \alpha A)^{-1} \mathbf{1}, \tag{14}$$

This can be calculated directly, taking care to choose a value of α below which the determinant diverges, which is where $\det(A - \alpha^{-1}\mathbf{I}) = 0$, i.e. $\alpha^{-1} = \lambda_1$, the largest eigenvalue of A, hence we must choose $\alpha < 1/\lambda_1$. A more efficient method is by an iterative process similar to the previous case:

$$\mathbf{x}' = \alpha A \mathbf{x} + \beta \mathbf{1}. \tag{15}$$

A useful generalisation for real-world systems, where some nodes have some intrinsic quality, would be to have separate β_i for each node, based on that property.

PageRank. A further extension useful in problems such as web-page rankings is to normalise the influence of outgoing links by the total out-degree. In this way highly ranked nodes would not give undue ranking to the many unimportant neighbours they cite:

$$x_i = \alpha \sum_j A_{ij} \frac{x_j}{k_j^{\text{out}}} + \beta, \tag{16}$$

artificially setting $k_j^{\text{out}} = 1$ for any which really have $k_j^{\text{out}} = 0$. This can be written:

$$\mathbf{x} = \alpha A D^{-1} \mathbf{x} + \beta \mathbf{1}, \tag{17}$$

where D is a diagonal matrix with:

$$D_{ii} = \begin{cases} 1 & \text{if } k_i^{\text{out}} = 0, \\ k_i^{\text{out}} & \text{otherwise.} \end{cases} \tag{18}$$

This can be solved to give:

$$\mathbf{x} = \beta (I - \alpha A D^{-1})^{-1} \mathbf{1}. \tag{19}$$

This centrality measure is called the *"PageRank"* (named after Larry Page of Google) and is used by Google to rank web-pages.

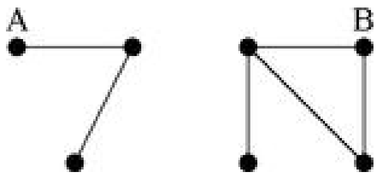

Figure 1: A network with two components.

4.3 Other spectral measures

Another thing eigenvalue of matrices representing the network structure can do is give information about the connectivity of the network.

A network with multiple disconnected components has an adjacency matrix that can be rearranged by swapping rows and columns to give a block diagonal matrix:

$$A = \begin{pmatrix} A_1 & \mathbf{0} & \cdots \\ \mathbf{0} & A_2 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix},$$

but it is not immediately obvious from the original unsorted A which nodes are in which component. The eigenvalues of a matrix related to the adjacency matrix can be used to give information on the number of components amongst other things.

4.4 The Graph Laplacian Matrix

Another useful matrix comes from studying diffusion on networks, where the flow into a *site* depends on the density difference between it and neighbouring sites. Other processes where the evolution of the state of a node has a linear dependence on the difference between its own state and that of its neighbours can also be modelled in similar ways, and diffusion processes are often used for modelling the spread of information or disease on networks. If each node i has an associated quantity x_i, and the flow from a neighbouring node j occurs at a rate $\sigma(x_j - x_i)$ where σ is a diffusion constant:

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \sum_j A_{ij}\sigma(x_j - x_i). \tag{20}$$

This can be rearranged to give:

$$\begin{aligned} \frac{\mathrm{d}x_i}{\mathrm{d}t} &= \sigma \sum_j a_{i,j}x_j - \sigma x_i \sum_j a_{i,j} \\ &= \sigma \sum_j a_{i,j}x_j - \sigma x_i k_i, \end{aligned} \tag{21}$$

since we have already seen that $A\mathbf{1} = \mathbf{k}$. This can then be rearranged into matrix form:

$$\dot{\mathbf{x}} = \sigma(A - D)\mathbf{x}, \tag{22}$$

Where D is a diagonal matrix with $\mathbf{k}$ on the diagonal and x is a vector of node states. The matrix:

$$L = D - A \tag{23}$$

is known as the *graph Laplacian*, and takes the place of the Laplacian operator for diffusion in continuous media ∇^2.

The graph Laplacian for a symmetric N node system is therefore an $N \times N$ matrix, with elements:

$$L_{(i \neq j)} = \begin{cases} k_i & \text{if } i = j, \\ -1 & \text{if } i \neq j \text{ and there is an edge } (i,j), \\ 0 & \text{otherwise,} \end{cases} \tag{24}$$

so that rows and columns all sum to zero.

For the network in Figure 1 B the graph Laplacian is:

$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix}$$

Equation 22 then becomes:

$$\dot{x} + \sigma L x = 0. \tag{25}$$

4.5 Eigenvalues and eigenvectors of the Laplacian

Since L is symmetric it has real eigenvalues, and it can also be shown that they are non-negative.

The *edge incidence matrix* B is a matrix similar to the one seen in the previous lecture, except that it is for simple 2-edges and elements take positive or negative values depending on which (arbitrarily chosen) end of an edge the vertices i and j are, i.e.:

$$B_{ij} = \begin{cases} +1 & \text{if end 1 of edge } i \text{ is attached to vertex } j, \\ -1 & \text{if end 2 of edge } i \text{ is attached to vertex } j, \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

It can be easily verified that the following is true:

$$L = B^T B. \tag{27}$$

Therefore, if $\mathbf{x}$ is a normalised eigenvector of L corresponding to eigenvalue λ_i:

$$\begin{aligned} L\mathbf{x_i} &= \lambda_i \mathbf{x_i}, \\ \mathbf{x_i^T} L \mathbf{x_i} &= \mathbf{x_i^T} \lambda_i \mathbf{x_i}, \\ \mathbf{x_i^T} B^T B \mathbf{x_i} &= \lambda_i \mathbf{x_i^T} \mathbf{x_i}, \\ (\mathbf{x_i^T} B^T)(B\mathbf{x_i}) &= \lambda_i. \end{aligned} \tag{28}$$

Since $B\mathbf{x_i}$ is a real vector, $\lambda_i \geq 0$.

4.6 The zero eigenvalues

The Laplacian always has at least one zero eigenvalue since:

$$L\mathbf{1} = 0, \tag{29}$$

due to the fact the rows all sum to zero.
For a graph with multiple components, which can be written in block diagonal form with the jth block being the Laplacian L_j of the jth component, it is easy to see that any eigenvector $\mathbf{x}$ with elements $x_i = 1$ for nodes in a component j and $x_i = 0$ elsewhere will also have a zero eigenvector:

$$\begin{pmatrix} L_1 & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & L_2 & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & L_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \end{pmatrix} = \mathbf{0}, \quad \begin{pmatrix} L_1 & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & L_2 & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & L_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \vdots \end{pmatrix} = \mathbf{0}, \quad \ldots$$

It can thus be shown that the number of zero eigenvalues is the number of components of the network. Due to this, the second eigenvalue λ_2 is called the *algebraic connectivity* and is only non-zero if the graph is connected.

4.7 Spectral graph partitioning

Things are less clear cut (literally) when there are elements connecting the different "components" (*communities*). It is not generally possible to cleanly divide the network into clear communities if there are many overlaps, but studying the eigenvectors of the Laplacian gives some indication of the structure.
The eigenvectors corresponding to larger, non-trivial, eigenvalues of L for a connected network are orthonormal and have a mixture of negative and positive elements. The graph can be partitioned based on the sign of these elements, and the eigenvector $\mathbf{x_2}$, corresponding to the second smallest eigenvalue λ_2, known as the *Fiedler vector*, is used to partition networks into two, where this is appropriate. This eigenvector is chosen because the cut-size can be shown to be proportional to the magnitude of the corresponding λ (See Newman Chapter 11).
Comparing n sequential eigenvectors against each other in an n dimensional space is found to be a useful method for partitioning graphs into communities, and often plotting $\mathbf{x_2}$ against $\mathbf{x_3}$ shows which nodes belong to which communities (with the minimum number of edges between them (the *cutsets*)), and how strongly these are separated. Measures based on the Euclidean distance between the vectors or components can be used to quantify the similarity or distance between communities and their nodes (See Boccaletti Section 7).

4.8 Betweenness partitioning

Other algorithms exist for partitioning graphs, including sequential edge-swapping and cutting followed by measuring the connectivity of the resulting clusters in an iterative manner.
One other method is to assigning betweenness scores to *edges*, using a similar method to that for nodes in in Section 4.2.1. Removing these *between community* links in sequence can result in partitioning networks into useful communities.

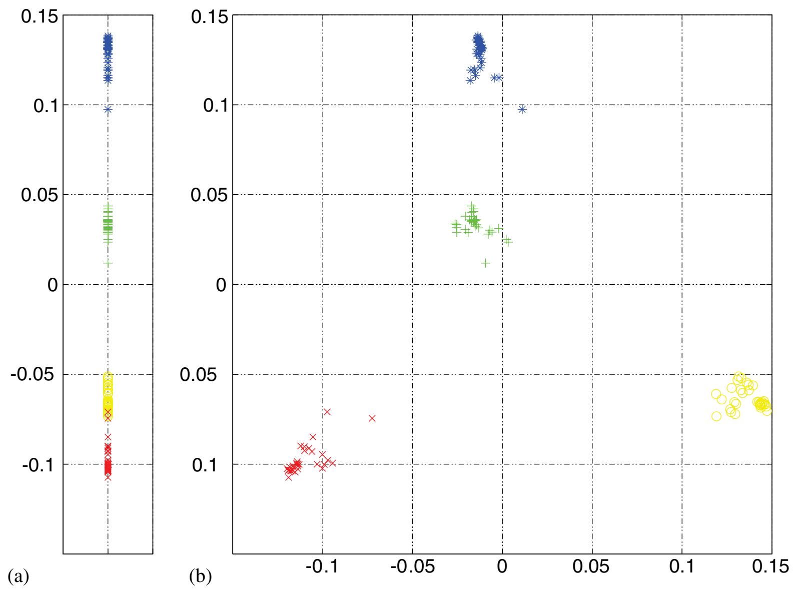

Figure 2: (a) Components of the first non-trivial eigenvector $\mathbf{x_2}$ for a computer-generated network with four communities. (b) All communities can be identified when the components of $\mathbf{x_3}$ are plotted versus those of $\mathbf{x_2}$ [Figure originally from L. Donetti & M.A. Muñoz, J. Stat. Mech. (2004)].

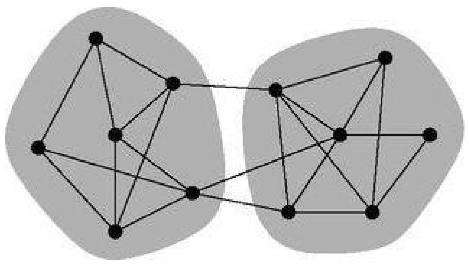

Figure 3: Connected *communities*.

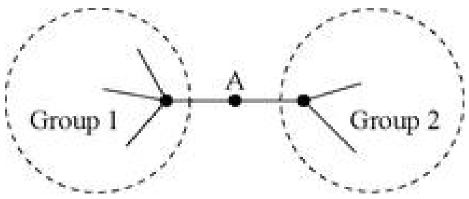

Figure 4: The node or edge at A has a high betweenness centrality, despite having few links, due to its importance connecting pats between different *communities* in the network.